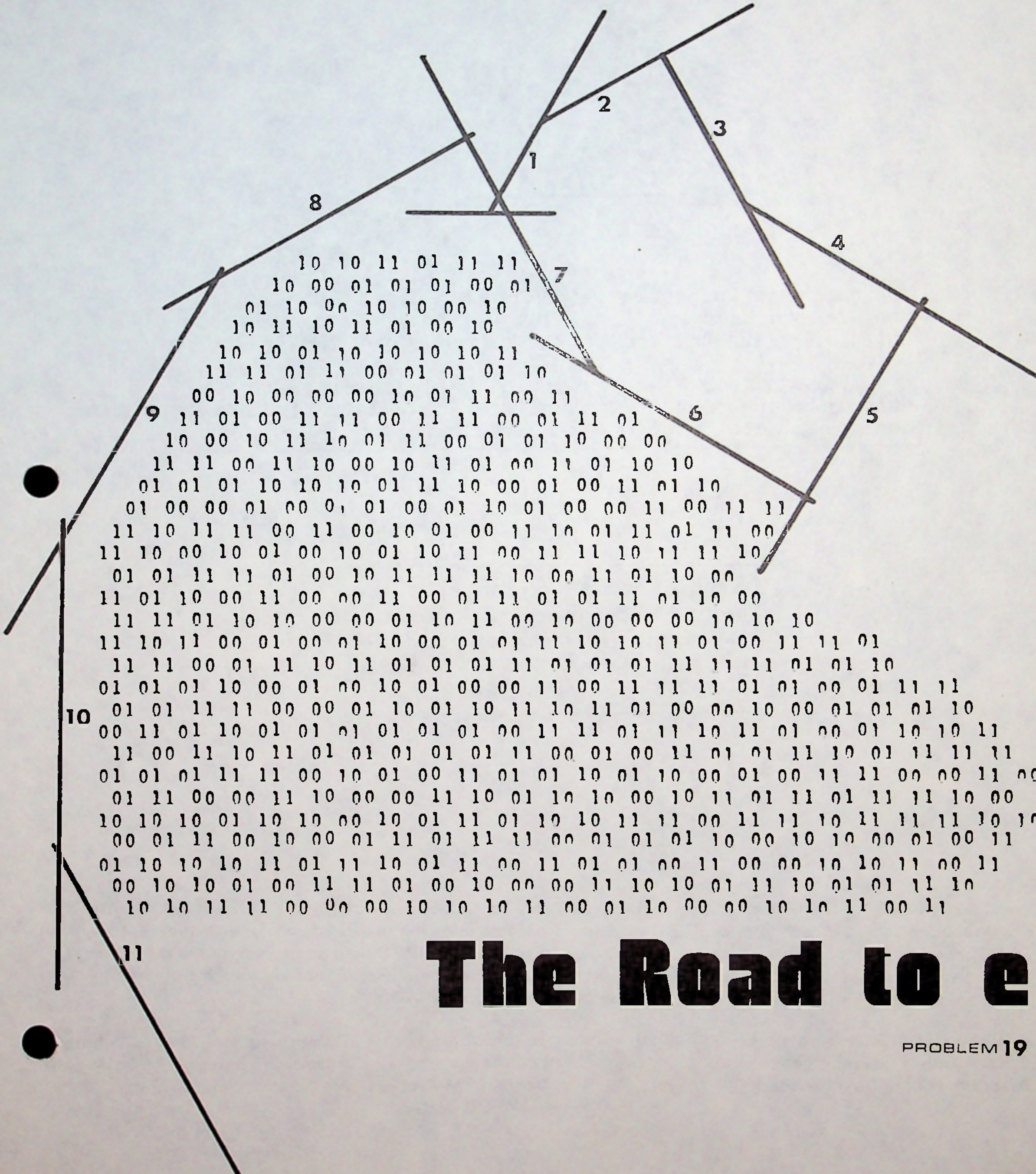
Popular Computing

BOX 272 CALABASAS, CA 91302

8

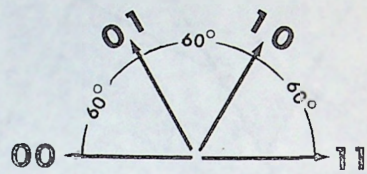
VOL 1
NO 8
NOV 73



The Road to e

PROBLEM 19

The number e (2.718281828... in decimal notation) is given on the cover in binary notation for its first 1000 bits. Also shown are the first eleven legs of a trip that is to be made as follows: each pair of bits gives 4 combinations, which are used to select a new direction according to the pattern:



The distance to be travelled on each leg is the square root of the number of that leg. The problem, then, is this: where is the end of the 500th leg of the Road to e?

The number e is one of the easiest series to calculate by computer; it has been calculated to 100,000 decimal places by Shanks and Wrench in 1961.

Elmer's Law: If the probability of a natural phenomenon having one of two stable states is .5 and you desire it to be in state A, then the probability is 1.0 that it is in state B.

POPULAR COMPUTING is published monthly at Box 272, Calabasas, California 91302. Subscription rate in the United States is $15 per year, or $12 if remittance accompanies the order. For all foreign subscriptions, add $4.50 per year. Multiple subscriptions to the same address, add $5 each; thus, 3 copies per month is $25 per year, U.S. delivery. Back issues $1.50 each. Subscriptions may begin with any issue. Subscriptions for qualified undergraduate students half price.

The following solution is furnished by Harry Nelson for Problem 10 (Three Track) from PC-5, page 6. The table shows that the paths of A and B cycle with length 16 moves, beginning at move 8. Thus, at move 1000, the position will be the same as at move 14. A reduces his gap behind B from 12 sectors after B's 4th move to 3 sectors after A's 16th move, but never gets any closer. "No computer was needed or used for the solution."

"A good problem which would probably need a computer is this: For each possible choice of A's initial sector and B's initial track and sector (called the first move),

1) Determine (if) when does A catch B.
2) Enumerate the cycle (or complete path)."

	A		B	
Move	Track	Sector	Track	Sector
1	1	4	3	16
2	2	6	3	18
3	2	9	2	19
4	1	10	1	22
5	2	13	1	24
6	2	14	2	25
7	2	16	1	26
8	1	18	3	27
9	3	20	1	29
10	2	23	2	1
11	3	26	1	4
12	2	29	1	6
13	2	1	3	9
14	2	4	2	11
15	2	6	3	12
16	2	9	2	13
17	1	10	1	15
18	2	13	1	17
19	1	15	2	20
20	3	16	2	23
21	2	17	3	26
22	1	18	3	27
.	.	.	.	.
.	.	.	.	.
1000	2	4	2	11

The following solutions have been received for Problem 12 (The Pi Dragon). Steven Stepanek ran the calculations in both Fortran and ALGOL, obtaining

-9.281152955
114.8455918

as the coordinates of the center of the 1000th pentagon. Harry Nelson, solving the same problem by hand calculation, obtained

-8.2811529493737
115.8970540297105

Daniel Shanks noted that at two places--from the 411th to the 420th places and from the 940th to the 950th places--there are runs of 10 and 11 odd digits, so that "the dragon goes through a complete circle and finds itself on the next sheet of the Riemann surface." The 10 even digits from the 70th through the 79th places perform similarly.

In the statement of the problem, PC6-2, the rotation shown was given as 54°, which should have been given as 36°.

PC2-9 presented a table of factorials; a corresponding table of subfactorials appeared in PC1-12. If corresponding entries from these tables are divided, the result appears to approach e, the natural logarithm base.

The following solution to Problem 11 (The 100 Square Trip) is furnished by Irwin Greenwald.

```
BEGIN
      q = Quotient (N/10)   ;
      r = Remainder (N/10)  ;
      c = c+q;
      IF q is odd THEN R = 11 - R  ;
      IF c is odd THEN R = R - r ELSE  R = R + r ;
      IF NOT (1 ≤ R ≤ 10) THEN
            BEGIN
                  c = c + 1 ;
                  IF R ≤ 0  THEN  R = |R| + 1
                            ELSE  R = 21 - R ;
            END
      IF c = 10  THEN  output (11,10)
                 ELSE  output (R,c) ;
END
```

In the statement of Problem 18, PC7-14, the inventory of the pieces of tile should read <u>three</u> 2 x 2 blocks.

ODDS AND ENDS

The September, 1973 issue of <u>Playboy</u> ran a 5-page article, "Math Goes Mini," discussing pocket calculators in general and reviewing 13 machines, ranging from the Rapidman 800 through the HP-45.

The July, 1973 issue of <u>Changing Times</u> carried a 3-page article on "Electronic Calculators." The article reviewed 23 machines in the $100 to $200 range of the four function type.

THE DISTRIBUTION OF NUMBERS--MATHEMATICAL THEORY

by R. W. Hamming

Bell Laboratories Murray Hill, New Jersey

The anomalous distribution of numbers

$$D(x) = (\log x)/(\log b)$$

arises not only from the approach through the distribution of all physical constants as well as the approach of asking what the processes of multiplication and division do to various distributions of input numbers; the distribution also arises from a purely mathematical approach. This mathematical approach begins with a theorem due to Weyl which states that if a is an irrational number (meaning not a fraction, but something like π, the square root of 2, the logarithm of 3, etc.) then the distribution of the fractional parts of the numbers

$$a,\ 2a,\ 3a,\ 4a,\ \ldots$$

is uniformly spread out in the interval 0 x 1.

We shall not try to prove this theorem here, but will merely consider a possible computer verification of it. How shall we pick a number that looks like an irrational number? Clearly, we can pick only a fixed number of digits, and hence it must automatically be a rational number (we regard the point as being before the first digit). We know that a rational number, when divided out, has some period of repetition--a period less than the size of the denominator since once well along in the division process if a remainder occurs a second time, then the division process must repeat itself--while an irrational number has no such period. Thus, we try to pick a sequence of digits, say in binary form, that appears to have no period, and we avoid numbers like

.101010...
.01001001001...
.001100110011...

We therefore try numbers like

.10100100011...

hoping that it is reasonably irrational in its behavior.

Starting with such a number we add it to itself and ignore any carry to the left. We again add the number, again add the number, etc., each time ignoring the overflow on the left. As an example, we take the binary number .10100111 and do 19 steps to get the following distribution of the first two digits (the example was not rigged)

digits	frequency
00	5
01	5
10	5
11	5

which experimentally verifies the theorem.

What has this to do with the distribution of mantissas? It is the fractional part of the exponent that determines the mantissa, while the integer part of the exponent determines the power of the base b of the number. To see more details, suppose that x is uniformly distributed in the interval $0 \leq x < 1$. In mathematical notation, the probability of seeing a number less than x is exactly x itself; that is, for a uniform distribution

$$\text{Prob.}\left\{t \leq x\right\} = x.$$

Now consider

$$y = b^x$$

$$\begin{aligned}\text{Prob.}\left\{y \leq a\right\} &= \text{Prob.}\left\{b^x \leq a\right\} \\ &= \text{Prob.}\left\{x \leq \log_b a\right\} \\ &= \log_b a\end{aligned}$$

(by the first equation above). Thus, if the exponent is uniformly distributed, then the mantissa has the distribution D(x) that we have been finding in other connections.

As an application of this theorem, consider the successive powers of some number, say a. That is, we look at the distribution of the mantissas of the numbers

$$a,\ a^2,\ a^3,\ a^4,\ \ldots$$

If $\log_b a$ is irrational, then by Weyl's theorem, the numbers will have the distribution

$$D(x) = (\log x)/(\log b).$$

It is well known that the Fibonacci numbers x_n defined by the difference equation

$$x_{n+1} = x_n + x_{n-1} \quad \text{with } x_0 = x_1 = 1$$

are given by $x_n = \dfrac{1}{\sqrt{5}}\left[\left(\dfrac{1+\sqrt{5}}{2}\right)^n - \left(\dfrac{1-\sqrt{5}}{2}\right)^n\right]$

From this we see that as n increases, x_n is closely approximated by the first term and must therefore satisfy the distribution D(x).

Recently, Epp has proved various extensions of Weyl's theorem. For example, if instead of adding the same number <u>a</u> at each step, on the nth step we add a number of the form

$$A + B \log n + e_n$$

where A and B are constants and e_n is a sequence of numbers approaching zero, then we again get a uniform distribution for the fractional parts of the sum. In this formulation, provided $B \neq 0$, no condition of irrationality on A is required.

In the particular case of e_n all zero and $A = 0$, $B = 1$, we get the fact that

$$\Sigma \ln(n) = \ln(n!)$$

is equidistributed. Applying our previous result, this means that n! must have the distribution D(x).

A still further generalization of Weyl's theorem states that if the sequence g_n has the asymptotic property that

$$\left| g_n / g_{n+1} \right| \sim \begin{cases} Kn^r & r \neq 0 \\ K \text{ irrational if } r = 0 \end{cases}$$

then the fractional parts of the sum of the g_n are uniformly distributed in the interval. The power of this last result is very great and shows that the distribution D(x) is to be expected in many mathematically occurring sequences of numbers; for example, the Bernoulli numbers (a program for calculating them was written by Lady Lovelace, the world's first programmer, to run on Babbage's projected computer).

EXERCISES

1. Consider 3^n ($n = 0, 1, \ldots, 100$) and examine the distribution of mantissas by examining the distribution of $n\log_2 3$.

2. Pick an "irrational number" and check Weyl's theorem using 1000 values and 32 intervals to group the values.

3. Check the first 200 Fibonacci numbers against $D(x)$.

4. Using the difference equation

$$x_{n+1} = 2x_n + x_{n-1} \qquad x_0 = 0,\ x_1 = 1$$

find the distribution of the first 100 numbers.

5. Check the distribution of $1/n!$

6. Check that $1/2 + 1/3 \ln(n)$ of Epp's first result is equidistributed (this is the same as $3\sqrt{n!}\,/2^n$ being like $D(x)$).

Card Dealing

PROBLEM **20**

A deck of cards in the following order:

AH AD 2H AC 3H 2D 4H QC 5H 3D 6H 2C 7H 4D 8H 8C 9H 5D
10H 3C JH 6D QH JC KH 7D AS 4C 2S 8D 3S 9C 4S 9D 5S 5C
6S 10D 7S KC 8S JD 9S 6C 10S QD JS 10C QS KD KS 7C

has this property: if dealt, one card out face up; the next card moved to the bottom of the deck; next card out; next card to the bottom of the deck, and so on, the result is the deck in systematic order. Other styles of play are conceivable. For example, one could deal two cards out, then one underneath, two cards out, and so on.

Given the ordering that should appear on the deal, and the procedure for dealing, write a program that will produce the initial ordering of the deck.

Book Review

THE ART OF COMPUTER PROGRAMMING,
VOL. 3: SORTING AND SEARCHING

by Donald E. Knuth
Addison-Wesley Publishing Company, $19.50.

Reviewed by W. C. McGee

In his album, That Was the Year That Was, mathematician Tom Lehrer got off this classic one-liner on the subject of precocious achievement:

"It's a sobering thought to realize that when Mozart was my age, he'd been dead two years."

Donald Knuth's Art of Computer Programming puts many computer professionals in much the same frame of mind vis-a-vis their own accomplishments. It's a sobering thought to realize that while most of us have barely managed to stay abreast of certain limited aspects of computer technology, Donald Knuth has produced three volumes of his seven-volume tour de force on computer programming, and is presumably well along on the remaining four volumes.

Sorting and Searching is the third volume of The Art of Computer Programming, comprising Chapter 5 (Sorting) and Chapter 6 (Searching). Previous volumes have been as follows:

Volume 1: Fundamental Algorithms
Chapter 1: Basic Concepts
Chapter 2: Information Structures

Volume 2: Seminumerical Algorithms
Chapter 3: Random Numbers
Chapter 4: Arithmetic

Knuth defines sorting as the process of establishing a total ordering among a set of records on the basis of values ("keys") which appear in the records. As is customary, he distinguishes between internal sorting, in which all records can be fit into main memory; and external sorting, in which auxiliary storage must be employed. He cites three reasons for sorting:

1) to bring together records having a common key value (this is the traditional meaning of sorting);

2) to place the records in two or more files in sequence on a common key so that records from different files can be matched on a single pass through the files (this is a common technique in sequential file maintenance); and

3) to facilitate searching a set of records, particularly when they have been printed for human consumption.

Searching is defined as the process of locating (determining the address of) a record containing one or more items with specified values. Knuth distinguishes two kinds of searching: searching on a single item or key, and searching on multiple items ("secondary key searching"). The need for searching follows simply from the need to access data which has been stored in a computer.

It is not possible in this limited space to do a comprehensive review of Knuth's latest volume. This certainly should be done, and presumably will be done in other publications. What we can do here is to indicate what we believe to be the major contributions of the volume, and to point out certain aspects of the volume which we feel will be of particular interest to readers of POPULAR COMPUTING.

Perhaps the most significant contribution of this volume is to provide a structure or framework for the knowledge which currently exists on the subjects of sorting and searching. As a teacher of these subjects, and as a contributor to their development, Knuth is clearly concerned about the welter of disorganized material which has accumulated on them, and strives in this volume to set matters straight. His most successful effort, in our opinion, comes in internal sorting. According to Knuth, all internal sorting techniques fall into one of the following five categories:

1) insertion
2) exchanging
3) selection
4) merging
5) distribution

Some of the better-known internal sorting techniques which are discussed in this framework are Shell's sort (insertion), bubble sort (exchanging), quicksort (exchanging), and tree selection (selection).

Similarly, Knuth provides the following categorization for single-item searching:

1) sequential searching
2) searching by (linearly ordered) key comparison
3) digital searching
4) hashing

Category (2) differs from category (1) in the respect that the key being searched on can be linearly ordered; this can produce significant economies in searching. Category (2) covers such well-known techniques as binary search as well as the indexing techniques found in many data management software packages.

Attempts to structure the areas of external sorting and multiple-item searching are less successful, perhaps because the body of knowledge on these subjects is not yet fully developed, or perhaps because they are not as mathematically tractable as internal sorting and single-item searching. Knuth nevertheless succeeds in providing a comprehensive survey of existing methods. There is one fold-out diagram in the section on external sorting that is particularly revealing: it is a set of charts depicting the time sequencing of major events (input, output, rewind, etc.) in some eight or nine different sorting methods, including such well-known methods as polyphase and cascade sort. For visual impact, this illustration has rarely been equalled (outside of Playboy).

Knuth's ability to impart structure to the complex subjects of sorting and searching is due in no small measure to his phenomenal grasp of the technical literature. Whereas most authors would be content to cite published works, Knuth frequently cites unpublished Ph.D. dissertations. Many of these he undoubtedly came across in his own work in the field (he has written papers on such topics as sorting networks and binary search techniques), and many were undoubtedly produced by his own students. Knuth also has the unusual ability of looking at a subject from an historian's viewpoint (he has written extensively on the history of mathematics, including articles on von Neumann's first computer program and ancient Babylonian algorithms). At several points in this book he interrupts the main stream of the exposition to retrace the same ground from an historical viewpoint. Not only does this serve to consolidate the material presented previously, but it also provides the opportunity to cite important work not previously referenced and to supply incidental but illuminating tidbits of information (such as the origin of the term "hash").

The second major contribution of Sorting and Searching is the new standard of excellence it establishes for the analysis of algorithms. Knuth makes it clear from the beginning that he is not so much interested in the subjects of sorting and searching for their own sake as he is in what they can teach us about the more general problem of analyzing complex algorithms. Knuth devotes considerable space at the beginning of the book to developing such mathematical concepts as permutations, inversions, and runs; he subsequently uses these concepts to systematically analyze various sorting and searching algorithms, for example, to estimate their average execution time and to develop optimum algorithms. It seems clear that the mathematical concepts he introduces and the way they are employed to analyze algorithms are considerably broader in scope than sorting and searching; if the reader grasps this, Knuth will have achieved one of his primary objectives.

Using one subject as an approach to learning another more general subject is just one sign of the teacher coming out in Knuth. Other such signs include the use of examples to introduce complex algorithms, and the use of analogies as an aid in understanding and analysis of algorithms. Two such analogies may be of particular interest to readers of POPULAR COMPUTING because of their resemblance to puzzles. The first analogy, due to E. F. Moore, is useful in analyzing the replacement selection technique for generating the initial strings in an external sort:

Snow is falling at a constant rate. A snowplow travels a circular road at a constant speed. If the total amount of snow on the road at any instant is P tons, how much snow must the plow remove on each circuit of the road in order that the total amount present remain constant? (What is the expected length of the runs produced by a P-way merge of random keys?)

The second analogy is due to R. M. Karp and is useful for analyzing the one-tape sorting problem:

A building has N floors and one elevator with a capacity of B people. There are C people on each floor waiting for the elevator, each person having a destination which is randomly selected subject only to the restriction that exactly C people are destined for each floor. What is the fastest way to transport these people to their destinations? (What is the fastest way to sort a tape of N blocks of C records each, if internal storage is limited to B records and external storage is limited to the given tape (which may be backspaced and rewritten)?)

The book abounds with exercises. As in the preceding volumes, the exercises are graded, from 00 (extremely easy) to 50 (research problem), and answers to all are given.

Finally, this book is fun to read. Sorting and searching are obviously subjects which Knuth has enjoyed working with and writing about, and he manages to convey to the reader a sense of his enjoyment at discovering the unobvious and unexpected. In addition, he refuses to take himself or his subject too seriously: signs of gentle humor appear unexpectedly in almost all parts of the book. It would not be fair to the prospective reader to quote these instances extensively, but one example might be appropriate just to convey the flavor of the humor in question. Following the discussion of the trie, a special kind of tree conceived by E. Fredkin to facilitate information retrieval, Knuth assigns the following exercise: "If a tree has leaves, what does a trie have?" (The exercise, incidentally, is rated 00).

Sorting and Searching is not easy to read, nor is it inexpensive to buy. The student willing to make the necessary investment of time and money, however, will find his investment more than amply repaid.

Book Review

THE CALCULATOR HANDBOOK, by A. N. Feldzamen and Faye Henle, copyright by Bowmar Instrument Corporation, Berkeley Medallion Book, May 1973, $1.25.

Despite the fact that Dr. Feldzamen is an experienced author (The Intelligent Man's Guide to Computers) and Miss Henle had a long career as a columnist, this is a poor book. It was apparently rushed into print, with a consequent lack of care for editing and proofing. The title is pure hyperbole.

The book's cover shows a Bowmar calculator with a 10-digit display, but the authors evidently worked exclusively with an 8-digit machine. Their use of it was curious; they report, for example, that

$$65536^2 = \%!\#*\&@!\#!!.$$

Every machine reviewed in PC2 page 5 will give, for that calculation, the result

42.949672

with an indication of overflow so that the true decimal point is 8 places to the right.

The bulk of the book describes problem situations for which the 4-function pocket calculator can be used: recipe doubling and halving; roofing; paneling; wallpaper; stocks and bonds; insurance; taxes; etc. Although they say "...its speed and capacity will permit you to calculate square roots, cube roots, and so on, with great ease...," only the Newton formula for square roots is shown.

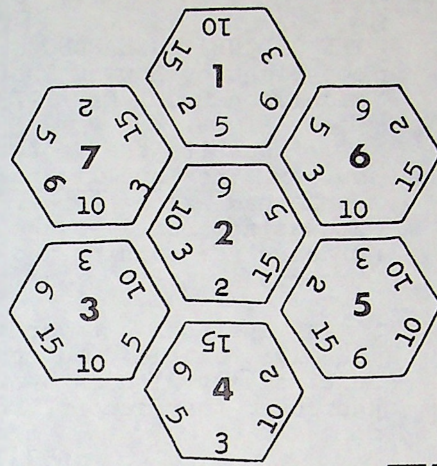
The chapter on "Selecting a Personal Calculator" recommends machines that have floating decimal, 8-digit capacity, battery operation, a clear-entry key, and a constant key; that is, the Bowmar machine.

Under "Diversions" there is given: "Enter [any integer] on the calculator, and multiply by 5. Add 6 to the product, multiply by 4, add 6.9, and then multiply by .05. Now subtract 1.65. You should have the number you started with. Try it." If you try it in the store, you won't buy a calculator.

The book concludes with 7 pages of conversion tables.

Anyone who intends to make more than casual use of an electronic calculator will find little of value in this book. The "Applications Guide" that Texas Instruments publishes (for their SR-10) is significantly more valuable.

Problem 9 Solution



A solution to Problem 9 (The Hexagon Problem, PC-5) comes from Thomas R. Parkin, of Control Data Corporation. The problem called for rearranging seven hexagons so that the numbers on adjacent edges would have no factor in common.

Mr. Parkin writes, "It seemed like such a nice example of backtracking that I couldn't forego the pleasure of solving it. Enclosed is...the solution method and the solution.

"Each configuration can be examined as was suggested in the problem, where failing any test implies a backup. The program for testing the configurations of the seven hexagons by successive rotations of individual figures, with appropriate backtracking when the configuration fails a test, is the inner loop of the program. If a particular configuration does not yield the solution, naturally the positions of the hexagons must be permuted and further tests made on each permutation. Only five of the hexagons on the rim need to be permuted, of course, since circular permutations of all six (considering the central figure fixed) would yield no different arrangements since the center hexagon can be rotated. To generate the order of permutations, naturally I used the Johnson-Trotter permutation algorithm, since that seems to be the cleanest and neatest, requiring only adjacent transpositions.

"After testing 120 permutations of the five figures on the rim (considering, say, hexagons 1 and 2 fixed), it is only necessary that each successive hexagon be tried in the center with any one other remaining fixed on the rim. In this way, it is necessary to try seven base configurations and 120 permutations on each, yielding no more than 840 cases to be tested for backtracking.

"I was in hopes that this problem might yield a few seconds of running time on the 6600, but unfortunately you drew the original picture with hexagon 2 as the center item. Naturally, pursuing iterations in the normal lexicographical order, the program disposed of the case with 1 in the center, then went on with the case of 2 in the center. Consequently, the program lasted only 1.314 seconds of computation time on the 6600 (in Fortran). Only 26 of the second permutations needed to be tried; thus, only 146 configurations were tested--approximately

nine milliseconds per test, including the time to permute the elements. Approximately 131 individual hexagon rotations and tests were needed for each configuration tested. The whole problem took 2.972 seconds, including 1.658 seconds for compilation and loading. The 6600 is quite fast.

"I ran my program through all possible cases (this took 6.561 seconds) to verify that there is only one solution. There were approximately 88000 individual hexagon rotations; the backtracking algorithm went up somewhat over 4000 times and down approximately 12000 times.

"I encoded each face as a particular pattern of three bits (one bit corresponding to the factor 2; one bit for the factor 3; and one bit for the factor 5). The test for no common factor was then the logical product of these two being zero."

Self Checking Digit

PROBLEM 21

For a decimal number, assume that a self checking digit is appended according to the following algorithm. The digits in positions 1, 3, 5,... (counting from the right) are doubled. The digits in the even numbered positions are taken as they stand. These results are crossfooted, and the sum is subtracted from the next higher multiple of ten; the unit's digit of the final result is the self checking digit. For example, for the number 1325579, the calculation would be:

$$\begin{array}{ccccccc} 1 & 3 & 2 & 5 & 5 & 7 & 9 \\ 2 + & 3 + & 4 + & 5 + & 10 + & 7 + & 18 = 49 \end{array}$$

$$50 - 49 = 1$$

Other examples are the following:

001234	4
000000	0
378524	5
387524	4
676767	0
767676	3
100200	5

If self checking digits are applied to consecutive numbers, they form a sequence. What is the cycle length of this sequence?

PROBLEM 22

Decimal Integer Series (DIS)

Consider a sequence of increasing integers. Each integer is changed to a decimal fraction by giving it twice as many decimal places as there are digits in the integer. The resulting fractions are then summed.

For those sequences in which the numbers "thin out" sufficiently rapidly, the process converges. For example, the sequence of squares:

.01
.04
.09
.0016
.0025
.
.
.0081
.000100

We will name the resulting numbers as follows:

X2	Squares
X3	Cubes (and so on for higher powers)
P2	Powers of 2
P3	Powers of 3 (and so on)
F	Fibonacci sequence
G	Factorials
SF	Subfactorials

with the following known results. Higher precision values, and other possible entries in this series are solicited. The values given here were calculated by H. P. Robinson.

X2	.18190 58902 00801 21567
X3	.10167 97738 27688 81770
X4	.02076 29270 97143 52162
P2	.15226 21564 48202 95983
P3	.13186 81318 78373 81835
P13	.01149 42527 52891 32896
F	.22372 23584 07324 78679
G	.09329 48357 09685 22600
SF	.12468 51718 16955 07735

The DIS process does not converge for the sequence of prime numbers. Any thinning out process that operated fast enough would lead to a convergent sum, such as:

a) Taking only the twin primes.
b) Taking only primes for which $p \pm 6$ is also a prime.
c) Taking only primes that do not contain a specific digit (e.g., 3).

Speaking of Languages

BY ROBERT TEAGUE

The subject this month is a short quiz on COBOL. The specific topic is DATA DIVISION entries--the OCCURS clause and the level structure specifically.

The following represents entries in the DATA DIVISION of a program.

```
01  A.
    02 X PIC S99 VALUE 2.
    02 Y PIC S99 VALUE 3.
    02 Z PIC S99 VALUE 4.
    02 AA.
       03 I PIC S99 VALUE 5.
       03 J PIC S99 VALUE 6.
    02 K PIC S99 VALUE 7.
01  B.
    02 J PIC S99 VALUE -2.
    02 BB.
       03 I PIC S99 VALUE 4.
       03 K PIC S99 VALUE 4.
    02 M PIC S99 VALUE 2.
    02 N PIC S99 VALUE 3.
    02 Z PIC S99 VALUE 4.
```

The statement below is part of the PROCEDURE DIVISION of the same program as the entries above.

```
MØVE CØRRESPØNDING A TØ B.
```

If the values of A and B were not changed prior to execution of the above MOVE, what will be the values possessed by the following entries in B: I, J, K, M, N, and Z?

The following entries are also part of a program's DATA DIVISION.

```
02 AA PIC X(36) VALUE 'ABCDEFGHIJKLMNØPQRSTUVWXYZ0123456789'.
02 BB REDEFINES AA.
  03 A ØCCURS 4 TIMES.
     04 B ØCCURS 3 TIMES.
        05 D PICTURE X.
     04 C ØCCURS 3 TIMES.
        05 E PICTURE X ØCCURS 2 TIMES.
```

What values do the following items possess at the start of execution?

D(2,2,2), E(2,2,2), B(3,3), C(3,1), A(4), E(2) IN C(3) IN A(1)

How many ways and how can the entry for E above be properly qualified with its subscripts?

One to Nine

PROBLEM 23

Prof. Richard Andree furnishes the following list of problems for computing students:

1. Find all integers I, J, K such that each lies between 2 and 5000 inclusive and

$$I^2 + J^2 = K^2 \pm 1.$$

2. Determine the smallest power of N such that N! contains

(a) all nine of the non-zero decimal digits;
(b) all ten decimal digits.

3. Determine the smallest exponent E_N such that N^{E_N} contains

(a) all nine non-zero decimal digits;
(b) all ten decimal digits,

for N = 2, 3, 4, 5, 6, 7, 8, 9 or show that E_N does not exist for certain of the specified values of N.

4. Which is larger: e^{π} or π^e?

5. $X^Y = X + Y = X \cdot Y$ has the obvious integral solution X = Y = 2. Are there other real solutions? Are there other complex solutions? Are you sure?

6. Determine the smallest positive prime divisor of $n^n - 1$ for n = 2, 3, ..., 10. If feasible, extend your table to n = 11, 12, ..., 25.

7. The number $(11826)^2 = 139854276$ contains each of the nine non-zero decimal digits exactly once, whereas $(99066)^2 = 9814072356$ contains all ten decimal digits each once. Find all perfect squares that have the property of containing all nine (or ten) decimal digits once and only once. What about cubes?

8. What is the largest integer which can be obtained as a product of several positive integers which add up to 100? If your answer is less than $.7*10^{15}$, try again.

9. Determine two rational fractions the sum of whose cubes is 6. (This problem apparently baffled the renowned 19th century French mathematician Legendre but was eventually solved by the English mathematician and puzzlist, H. E. Dudeney.)

The Way to Learn Computing is to Compute